LEGIBILITY NOTICE

A major purpose of the Technical Information Center is to provide the broadest dissemination possible of information contained in DOE's Research and Development Reports to business, industry, the academic community, and federal, state and local governments.

Although a small portion of this report is not reproducible, it is being made available to expedite the availability of information on the research discussed herein.

CONF-880455--3

TITLE: Rapid Prototyping of Simulations in Artificial Intelligence Environments

LA-UR--87-4255

DE88 004295

AUTHOR(S): Ron Martinez

SUBMITTED TO: Eastern Simulation Conference

MASTER

DISCLAIMER

Los Alamos Los Alamos National Laboratory
Los Alamos, New Mexico 87545

Rapid Prototyping of Simulations in Artificial Intelligence Environments

by
Ron Martinez
Simulation and Software Development Group
Los Alamos National Laboratory

Abstract

The benefits derived from rapidly constructing a prototypical simulation model for use in characterizing the scope of a project should not be underestimated. The initial period of contact with the end-user of a simulation will establish the 'ground-rules' by which the project will progress. This paper describes some experiences in the application of an expert system shell in the development of knowledge-based discrete event simulations. This model development approach leverages the benefits of object-oriented programming, frames for representing the objects to be simulated, and the graphics capabilities inherent within many expert system shells. The user interface relies heavily upon the use of graphical active images for the modification of important object attributes prior to runtime. Having a functional knowledge-based simulation facilitates the process of accurately determining the needs of the client.

1. Introduction

One of the most challenging and difficult aspects of a modeler's job is building an accurate model and convincing the end users that the model is a meaningful and accurate representation of the real system. Building a credible relationship with the client (as well a eliciting additional information about the system to be simulated) may be accomplished through the development of a prototype model early in the life of the project. Involving the client during the development of the prototype (and thus the process of defining the system to be simulated) creates a team effort which maximizes the likelihood that the model will be accepted and used in the decision making process. The client's acceptance and use of the model is the final test of its credibility; a prototype builds the initial foundation for establishing this credibility.

For the ultimate credibility of the model, it is very important from the outset that the modeler not raise expectations too high. Knowledge-based simulation alone will not solve problems, (just as Artificial Intelligence tools and techniques will not produce miracles) but it does provide a valuable tool for identifying problem areas and testing alternative solutions. In some cases, the use of AI tools and techniques will make the client question the chances for success of the development process. Users want results, not the chance to patronize a high risk technology. The point must be stressed that these programming paradigms open new possibilities for building powerful systems.

It has been my experience that there is seldom a single individual who understands how a given system works in sufficient detail to build an accurate simulation model. In addition, simulation of some systems may require interacting with individuals from varied backgrounds and thus the required information may be difficult to extract. The modeler must have investigative skills and 'people' skills to

elicit the necessary information from the client's organization. It is invaluable to have an advocate among the client's staff who will dig out the answers to detailed questions concerning system operation or identify the appropriate individuals to be contacted.

In order to establish momentum for the project, get a simple model up and running as quickly as possible, and then later embellish it. This is a good way to generate and maintain the client's interest and involvement. Strike a balance between the level of detail needed to achieve the objectives and the level needed to convince the client of the model's validity. The implementation of a prototype also serves to provide a forum for all participants and thus insures that common concepts are being discussed. The prototyping period serves to build a common understanding concerning the assumptions which will be built into the model.

In developing a prototype, using terminology specific to the application area facilitates mutual understanding. It has been my experience that there can be a different conceptual understanding of many of the system characteristics, further confusion of the issue with a computer scientist's terminology does not help.

The prototyping phase will help identify (as early on as possible) the data that will be needed for the full scale simulation.

2. Prototyping Guidelines

When implementing a prototype, there are a number of guidelines to follow which serve to leverage the prototype's value. They are:

(1) Use application terminology for simulation entities and functionality

(2) Monitor appropriate simulation entities

(3) Use graphical user interfaces

(4) Provide animation if possible

Users feel most comfortable dealing with concepts, terminology and units of measurement with which they are familiar. A user who is used to thinking about velocity in 'miles per hour' will not like making the conversion to 'meters per second'. Similarly, the 'terminal phase of a ballistic trajectory' should not be referred to as the 're-entry phase'. The introduction of new concepts such as 'frames, forward chaining, methods, active values, inheritance, etc' makes the uninitiated client uneasy. Remember that the client is footing the bill for the project; alienating the client through the use of computing 'jargon' will not increase your credibility.

The use of a trace or monitoring capability for appropriate entity attributes allows the clients to satisfy themselves that the baseline functionality of the simulation entities is correct. For example, the monitoring of the status (location, etc.) of a simulation entity allows one to dynamically follow the course of events that influence the entity, thus providing the opportunity for the client to verify expected behavior. The monitoring of entity attributes is also very useful for debugging during the development phase.

A user interface which minimizes the users' need to understand the operating system syntax will be met with favor. IntelliCorp's Knowledge Engineering Environment (KEE) provides a means to accomplish this easily through the use of 'active images'; these active images are graphical displays for use in viewing and modifying attribute values within simulation objects. The graphical interface is intended to allow the user to easily manipulate the various parameters that drive the model. The user need only rely on the use of the mouse in pointing at various active images (attached to the attributes within various simulation entities). Thus, the end-user is not burdened with the need to understand the intricacies of the software environment in an attempt to use the model.

The use of animation removes many of the 'black-box' aspects of a simulation model and allows users to 'see' model assumptions in action rather that depending on the modeler's assurance and on

long run statistical output to verify model correctness. A model with high credibility can be achieved through an implementation that appears reasonable on its surface to the end users of the model.

The development of a prototype should serve to elucidate the details and scope of the simulation effort. A prototype can be 'thrown away' upon initiation of the formal coding phase, or serve as the basis for a stepwise refinement in developing the full model. Many people make the mistake of using the prototype as the starting point for the coding phase and thus limitations (inefficiencies) will be intermixed with the large simulation model. Take advantage of the chance to 'start from scratch' with the increased knowledge you have gained during the prototyping phase.

3. Discrete Event Prototyping in KEE

Discrete event simulation using an Artificial Intelligence environment benefits greatly from the use of special features within these environments. A knowledge-based discrete event simulation is the implementation of a simulation within an expert system shell utilizing the knowledge-base representation capability for describing the simulation entities. The model development approach leverages the benefits of object-oriented programming, frames for representing the objects to be simulated, and the graphics capabilities inherent within the expert system shell.

The object-oriented programming environment allows one to combine the attributes of procedures and data. Objects store data in variables (slots) and respond to messages by carrying out procedures (methods). A message is the specification of an operation to be preformed on an object, similar to a procedure call. The operations that an object can perform are defined by the set of methods that are specified within the slot structure. These methods are functions that implement the response when a message is sent to the object, an event is implemented as a method. The interaction between objects takes place by way of sending a message and receiving a response to the message.

Frame-based representation is a means of representing objects and their attributes. Typically, a frame describes a class of objects, with each frame consisting of a collection of slots that describe aspects of the objects. Objects that share similar instance variables (slots) and methods can have the common characteristics decomposed into a class hierarchy. The class is an object that describes what common characteristics are shared by either subclasses or instances of the class. The classes form a hierarchy of class-subclass relationships with instances as the leaves of the hierarchy. The ability to pass along attributes (slots) from class to subclass to instances of the classes is called inheritance.

Within the context of the simulation model and the KEE environment, objects are units. These units have slots describing their attributes, and which are related in a way that allows inheritance of attributes. The capabilities of each simulation entity are represented by methods. Each method is a body of lisp code which implements the functionality of the the desired behavior of the simulation entity.

The matching of a discrete event simulation with an object-oriented programming environment which allows for a frame-based representation allows one to concentrate on the simulation application rather than the details of implementation. Prototyping is faster, and the ability of the non-programmer to understand the application is enhanced, thus making for a better client relationship.

4. Prototyping Experiences

I will describe two experiences involving prototyping, and discuss the qualitative impact on each project. The first case involved the simulation of a Vehicle Survivability System (VSS), the second involved the simulation of the money laundering process as it impacts various domestic and international institutions.

4.1. VSS Prototype

The VSS concept can be described as an active point defense for ground combat vehicles which functions independently of the activities of the combat vehicle itself. The VSS would neutralize

incoming warheads being targeted at the combat vehicle hosting the VSS. The incoming warhead could be one of a host of threats; the rocket propelled grenade, a cannon launched high explosive, etc. The VSS is composed of:

(1) A single 'detection system' which is composed of a radar system coupled with a CPU. The CPU performs the task of interfacing with the radar system, preliminary data formatting and interpretation, threat flight path calculations, and ultimately performs engagement calculations designed for use in driving the counter-munition system.

(2) A counter-munition system for targeting an incoming threat projectile.

The prototype was developed with the capability of representing two-dimensional movement of the combat vehicles, with the engagement calculations taking into account the direction of movement as well as the velocity of the target vehicle in calculating the impact point of the threat round.

The user interface was implemented using KEE active images to facilitate the initiation of the simulation, to monitor appropriate attributes, and to provide a means for modifying input parameters. The status of the combat vehicles is continually depicted using bitmap representations of two tanks which portray the tanks in various modes; firing at the tank with a VSS, scanning for incoming threat projectiles, counter-firing, and getting hit by a projectile. Thus the user can easily see what is occurring during simulation without having to closely monitor the event trace.

Figure 1. shows the user interface for the VSS prototype.

Figure 1. VSS User Interface

The prototype was very well received within the VSS multi-disciplinary project team, the team members were not 'computer wizards' and appreciated the fact that the model was easy to understand and more importantly - easy to use. I have since been tasked with the generation of more detailed simulation models for four anticipated engineering approachs to VSS as a result of the prototype. Initial results from the first detailed model were well received.

4.2. Money Laundering Prototype

The 'money laundering' process is the transformation of large amounts of illegally acquired bulk currency into what appears to be legitimate financial assets. Bulk currency is converted into some type of financial instrument using a variety of schemes to avoid the detection of this infusion of large amounts of cash into the financial system. The process typically involves the use of both domestic and international business interests and financial institutions.

The prototype required the explicit representation of the active organizations, financial as well as those involved in the illegal activities which require their profits to be laundered. The dynamic nature of money laundering would be represented (and tracked) at each simulation entity (actor) as the funds flow through a number of postulated money laundering schemes. A money laundering scheme could involve any number of simulation entities, financial institutions as well as money laundering actors.

The approach taken was to represent the business and financial system as a large 'network' of closely coupled institutions, each having the ability of interacting with a (potentially) large suite of financial actors. This network of institutions made extensive use of inheritance in describing those attributes that were common at each level of the hierarchy. In a departure from the first prototype application, the use of KEE's rule system for representation of the money laundering operational guidelines played an important role. The operational guidelines were represented as a set of decision rules which could be invoked from within events. These rules can be expressed in a manner that can be easily understood (and modified) by the user. An example of a decision rule for a money laundering actor could be:

```
IF the ethnic-background of the ?suspect is 'columbian'
   and
   the ethnic-background of the ?money-laundering-agent is 'columbian'
THEN
   the money-laundering-agent of ?suspect is ?money-laundering-agent
```

The way in which the suite of simulation entities interact during the course of the simulation was intended to allow for the analysis of the cause and effect relationships of many simultaneous money laundering suppression schemes upon the financial infra-structure. The dynamic nature of money laundering would be represented (and tracked) at each simulation entity as the funds flow through a number of postulated money laundering schemes. The prototype was developed to provide a proof of concept regarding the ability of the simulation approach to address the problem.

The project floundered in its early stages due to the fact that a broad variety of personnel were involved during the attempt to formulate the problem. Economists, attorneys, enforcement field agents: all had their own conceptual understanding of the problem as well as the approach to be taken in addressing the problem. The development of the prototype helped to create a common perception of the solution methodology, and provide the basis for communication. The project has not been funded at this point but the methodology has been favorably received.

5. Summary

A hybrid expert system programming environment which incorporates a variety of AI paradigms facilitates the development of prototypes in many application areas. The use of application terminology, appropriate monitoring of entity attributes, and graphically oriented user interfaces maximize the value of the prototype. Leveraging the features of an expert system programming environment in the development of a knowledge-based simulation provides the ability to quickly develop a good client relationship. Prototyping provides the ability to explore many questions about a system's performance that otherwise would entail the development of engineering prototypes (or at a minimum, the development of a large simulation model).

martinez@bellman

scs.paper

Thu Dec 17 10:22:39 1987

Sun LaserWriter

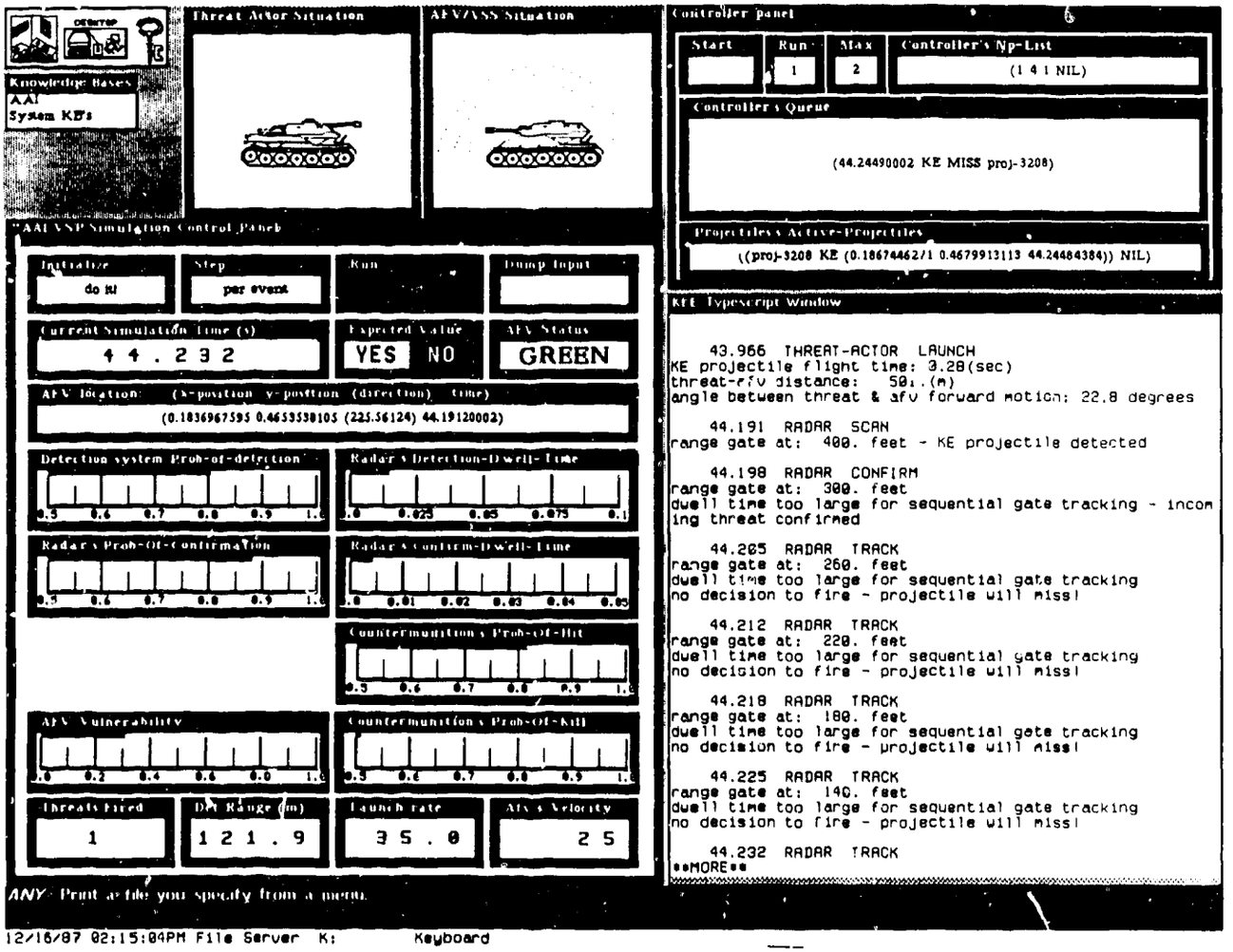
Sun LaserWriter bellman:martinez Job: scs.paper Date: Thu Dec 17 10:22:39 1987

Sun LaserWriter bellman:martinez Job: scs.paper Date: Thu Dec 17 10:22:39 1987

Sun LaserWriter bellman:martinez Job: scs.paper Date: Thu Dec 17 10:22:39 1987

Sun LaserWriter bellman:martinez Job: scs.paper Date: Thu Dec 17 10:22:39 1987

Sun LaserWriter bellman:martinez Job: scs.paper Date: Thu Dec 17 10:22:39 1987



12/16/87 02:15:04PM File Server K: Keyboard